**Хранимая процедура** – это поименованный код на языке Transact-SQL. Хранимая процедура может быть создана с помощью CREATE, изменена с помощью ALTER и удалена с помощью оператора DROP. Процедура может принимать входные и формировать выходные параметры. Результатом ее выполнения может быть целочисленное значение, которое возвращается к точке вызова оператором RETURN, либо один или более результирующих наборов, сформированных операторами SELECT, либо содержимое стандартного выходного потока, полученного при выполнении операторов PRINT.

Вызов процедуры осуществляется оператором EXECUTE (EXEC). В хранимых процедурах допускается применение основных DDL, DML и TCL-операторов, конструкций TRY/CATCH, курсоров, временных таблиц.

**Процедура для подсчета количества строк в таблице Заказы и вывода ее содержимого:**

Процедура принимает два параметра: входной параметр `@p` и выходной параметр `@c`. Процедура формирует значение выходного параметра `@c`, выводит в стандартный поток сообщение (`PRINT`), а также формирует результирующий набор. Вызов процедуры осуществляется оператором `EXEC`:

```sql

CREATE PROCEDURE PrZakazy

@p NVARCHAR(50),

@c INT OUTPUT

AS

BEGIN

SELECT @c = COUNT(\*) FROM Заказы WHERE Заказчик = @p;

PRINT 'Количество строк в таблице Заказы для заказчика ' + @p + ': ' + CAST(@c AS VARCHAR(10));

SELECT \* FROM Заказы WHERE Заказчик = @p;

END;

```

Вызов процедуры:

```sql

DECLARE @count INT;

EXEC PrZakazy @p = 'Zte', @c = @count OUTPUT;

```

\*\*Изменение процедуры с помощью ALTER:\*\*

С помощью оператора `ALTER` можно изменить процедуру. Пример изменения процедуры `PrZakazy`:

```sql

ALTER PROCEDURE PrZakazy

@p NVARCHAR(50),

@c INT OUTPUT,

@newParam INT

AS

BEGIN

-- Новый код процедуры с использованием нового параметра

END;

```

\*\*Использование временной таблицы и оператора INSERT:\*\*

```sql

-- Создание временной таблицы

CREATE TABLE #Zk (/\* описание структуры временной таблицы \*/);

-- Заполнение временной таблицы с использованием процедуры

INSERT #Zk EXEC PrZakazy @p = 'Zte';

-- Просмотр содержимого временной таблицы

SELECT \* FROM #Zk;

```

\*\*Процедура вставки строки в таблицу Товары с обработкой ошибок:\*\*

```sql

CREATE PROCEDURE TovaryInsert

@t NVARCHAR(50),

@cn INT,

@kl INT

AS

BEGIN

BEGIN TRY

INSERT INTO Товары (Наименование, Себестоимость, Количество) VALUES (@t, @cn, @kl);

END TRY

BEGIN CATCH

PRINT 'Ошибка: ' + ERROR\_MESSAGE();

-- Дополнительная логика обработки ошибок

END CATCH

END;

```

\*\*Процедура вывода отчета о товарах, заказанных конкретным заказчиком:\*\*

```sql

CREATE PROCEDURE Zkz\_REPORT

@p CHAR(50)

AS

BEGIN

DECLARE @rc INT = 0;

BEGIN TRY

DECLARE @tv CHAR(20), @t CHAR(300) = ' ';

DECLARE ZkTov CURSOR FOR

SELECT Наименование\_товара FROM Заказы WHERE Заказчик = @p;

IF NOT EXISTS (SELECT Наименование\_товара FROM Заказы WHERE Заказчик = @p)

RAISEERROR('Ошибка', 11, 1);

ELSE

OPEN ZkTov;

FETCH ZkTov INTO @tv;

PRINT 'Заказанные товары';

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @t = RTRIM(@tv) + ', ' + @t;

SET @rc = @rc + 1;

FETCH ZkTov INTO @tv;

END;

PRINT @t;

CLOSE ZkTov;

RETURN @rc;

END TRY

BEGIN CATCH

PRINT 'Ошибка в параметрах';

IF ERROR\_PROCEDURE() IS NOT NULL

PRINT 'Имя процедуры: ' + ERROR\_PROCEDURE();

RETURN @rc;

END CATCH;

END;

```

Вызов процедуры:

```sql

DECLARE @rc INT;

EXEC @rc = Zkz\_REPORT @p = 'Луч';

PRINT 'Количество товаров = ' + CAST(@rc AS VARCHAR(3));

```

\*\*Процедура добавления информации в таблицу Заказы и Товары:\*\*

```sql

CREATE PROCEDURE TovaryInsert\_X

@a INT,

@b NVARCHAR(50),

@c REAL,

@d INT = NULL,

@e DATE,

@f NVARCHAR(50)

AS

BEGIN

DECLARE @rc INT = 1;

BEGIN TRY

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

BEGIN TRAN

INSERT INTO Заказы (Номер\_заказа, Наименование\_товара, Цена\_продажи, Количество, Дата\_поставки, Заказчик)

VALUES (@a, @b, @c, @d, @e, @f);

EXEC @rc = TovaryInsert @b, @c, @d;

COMMIT TRAN;

RETURN @rc;

END TRY

BEGIN CATCH

PRINT 'Номер ошибки: ' + CAST(ERROR\_NUMBER() AS VARCHAR(6));

PRINT 'Сообщение: ' + ERROR\_MESSAGE();

PRINT 'Уровень: ' + CAST(ERROR\_SEVERITY() AS VARCHAR(6));

PRINT 'Метка: ' + CAST(ERROR\_STATE() AS VARCHAR(8));

PRINT 'Номер строки: ' + CAST(ERROR\_LINE() AS VARCHAR(8));

IF ERROR\_PROCEDURE() IS NOT NULL

PRINT 'Имя процедуры: ' + ERROR\_PROCEDURE();

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

RETURN -1;

END CATCH;

END;

```

Вызов процедуры:

```sql

DECLARE @rc INT;

EXEC @rc = TovaryInsert\_X @a = 20, @b = 'Стол', @c = 78, @d = 10, @e = '01.12.2014', @f = 'Луч';

PRINT 'Код ошибки = ' + CAST(@rc AS VARCHAR(3));

```